

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Shigehiro ASANO, et al.

GAU: 2185

SERIAL NO: 09/989,028

EXAMINER:

FILED November 21, 2001

FOR: MULTIPROCESSOR SYSTEM AND CONTROL METHOD THEREOF

REQUEST FOR PRIORITY

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON, D.C. 20231

RECEIVED

MAR 01 2002

Technology Center 2100

SIR:

- ☐ Full benefit of the filing date of U.S. Application Serial Number [US App No], filed [US App Dt], is claimed pursuant to the provisions of 35 U.S.C. §120.
- ☐ Full benefit of the filing date of U.S. Provisional Application Serial Number , filed , is claimed pursuant to the provisions of 35 U.S.C. §119(e).
- ☒ Applicants claim any right to priority from any earlier filed applications to which they may be entitled pursuant to the provisions of 35 U.S.C. §119, as noted below.

In the matter of the above-identified application for patent, notice is hereby given that the applicants claim as priority:

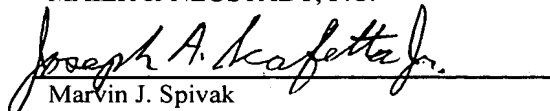
<u>COUNTRY</u>	<u>APPLICATION NUMBER</u>	<u>MONTH/DAY/YEAR</u>
JAPAN	2000-356237	November 22, 2000

Certified copies of the corresponding Convention Application(s)

- ☒ are submitted herewith
- ☐ will be submitted prior to payment of the Final Fee
- ☐ were filed in prior application Serial No. filed
- ☐ were submitted to the International Bureau in PCT Application Number .
Receipt of the certified copies by the International Bureau in a timely manner under PCT Rule 17.1(a) has been acknowledged as evidenced by the attached PCT/IB/304.
- ☐ (A) Application Serial No.(s) were filed in prior application Serial No. filed ; and
(B) Application Serial No.(s)
 - ☐ are submitted herewith
 - ☐ will be submitted prior to payment of the Final Fee

Respectfully Submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.


Marvin J. Spivak

Registration No. 24,913

Joseph A. Scafetta, Jr.
Registration No. 26,803



22850

Tel. (703) 413-3000
Fax. (703) 413-2220
(OSMMN 10/98)

09/989,028



日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2000年11月22日

出 願 番 号

Application Number:

特願2000-356237

出 願 人

Applicant(s):

株式会社東芝

RECEIVED

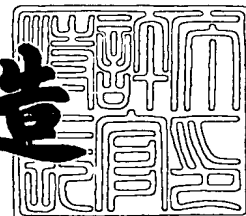
MAR 01 2002

Technology Center 2100

2001年12月21日

特 許 庁 長 官
Commissioner,
Japan Patent Office

及 川 耕 造



出証番号 出証特2001-3110915

【書類名】 特許願

【整理番号】 A000006367

【提出日】 平成12年11月22日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 15/00

【発明の名称】 マルチプロセッサシステムおよびその制御方法

【請求項の数】 7

【発明者】

 【住所又は居所】 神奈川県川崎市幸区小向東芝町1番地 株式会社東芝研
 究開発センター内

 【氏名】 浅野 滋博

【発明者】

 【住所又は居所】 神奈川県川崎市幸区小向東芝町1番地 株式会社東芝マ
 イクロエレクトロニクスセンター内

 【氏名】 斎藤 光男

【特許出願人】

 【識別番号】 000003078

 【氏名又は名称】 株式会社 東芝

【代理人】

 【識別番号】 100058479

 【弁理士】

 【氏名又は名称】 鈴江 武彦

 【電話番号】 03-3502-3181

【選任した代理人】

 【識別番号】 100084618

 【弁理士】

 【氏名又は名称】 村松 貞男

【選任した代理人】

 【識別番号】 100068814

【弁理士】

【氏名又は名称】 坪井 淳

【選任した代理人】

【識別番号】 100092196

【弁理士】

【氏名又は名称】 橋本 良郎

【選任した代理人】

【識別番号】 100091351

【弁理士】

【氏名又は名称】 河野 哲

【選任した代理人】

【識別番号】 100088683

【弁理士】

【氏名又は名称】 中村 誠

【選任した代理人】

【識別番号】 100070437

【弁理士】

【氏名又は名称】 河井 将次

【手数料の表示】

【予納台帳番号】 011567

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 マルチプロセッサシステムおよびその制御方法

【特許請求の範囲】

【請求項1】 マスタプロセッサと、ローカルメモリを各々が有し前記マスタプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレメントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムにおいて、

前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサエレメントのローカルメモリとの間のDMA転送を行うDMA実行手段と、

前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように前記DMA実行手段および前記各プロセッサエレメントにそれぞれ対応して設けられたコマンド蓄積用バッファと、

発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知する未応答コマンド数管理手段とを具備することを特徴とするマルチプロセッサシステム。

【請求項2】 前記マスタプロセッサ上で並列実行される複数のスレッドそれぞれに固有の識別子を用いて、前記各コマンドをその発行元のスレッドの識別子と共に前記マスタプロセッサから前記DMA実行手段または前記各プロセッサエレメントに発行する手段と、

前記発行されたコマンドに対応する応答を該当するスレッドの識別子と共に前記DMA実行手段または前記各プロセッサエレメントから前記マスタプロセッサに発行する手段とをさらに具備し、

前記未応答コマンド数管理手段は発行済みで応答が返ってきてないコマンドの数を前記スレッド毎に個々に管理し、すべての発行済みのコマンドに対して応答が返ってきたスレッドに対してコマンド処理の完了を通知するように構成されていることを特徴とする請求項1記載のマルチプロセッサシステム。

【請求項3】 マスタプロセッサと、ローカルメモリを各々が有し前記マス

タプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレメントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムにおいて、

前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサエレメントのローカルメモリとの間のDMA転送を行うDMA実行手段と、

前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように前記DMA実行手段および前記各プロセッサエレメントにそれぞれ対応して設けられたコマンド蓄積用バッファと、

発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知する手段と、

前記マスタプロセッサ上で並列実行される複数のスレッドそれぞれに固有の識別子を用いて、前記各コマンドをその発行元のスレッドの識別子と共に前記マスタプロセッサから前記DMA実行手段または前記各プロセッサエレメントに発行する手段と、

前記発行されたコマンドに対応する応答を該当するスレッドの識別子と共に前記DMA実行手段または前記各プロセッサエレメントから前記マスタプロセッサに発行する手段と、

前記複数のスレッドそれぞれの識別子を用いて、前記ローカルメモリの各記憶領域毎にどのスレッドに対応するプログラムまたはデータが格納されているかを管理する管理テーブルとを具備し、

前記DMA実行手段および前記各プロセッサエレメントは、それぞれ対応するコマンド蓄積用バッファに蓄積されている前記各スレッドからのコマンドの中で、実行可能なコマンドを前記管理テーブルを参照して特定し、その特定したコマンドを実行することを特徴とするマルチプロセッサシステム。

【請求項4】 前記各プロセッサエレメントは、前記管理テーブルを参照することにより、対応するコマンド蓄積用バッファに蓄積されている各スレッドからのコマンドの中で、ローカルメモリ内に該当するスレッドの処理対象データが

用意されたコマンドから順に処理を開始することを特徴とする請求項 3 記載のマルチプロセッサシステム。

【請求項 5】 前記管理テーブルは、前記ローカルメモリ内に前記プロセッサエレメントのコマンド処理で得られた処理結果のデータが用意されたか否かをスレッド毎に示す情報を含み、

前記 DMA 実行手段は、前記管理テーブルを参照することにより、コマンド蓄積用バッファに蓄積されている前記各スレッドからのコマンドの中で、前記ローカルメモリ内に該当するスレッドの処理結果のデータが用意されたコマンドから順に処理を開始して、前記ローカルメモリから前記グローバルメモリへの DMA 転送を行うことを特徴とする請求項 4 記載のマルチプロセッサシステム。

【請求項 6】 マスタプロセッサと、ローカルメモリを各々が有し前記マスタプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレメントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムの動作を制御するための制御方法であって、

前記マルチプロセッサシステムには、前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサエレメントのローカルメモリとの間の DMA 転送を行う DMA 実行手段、及び前記 DMA 実行手段および前記各プロセッサエレメントにはそれぞれ前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように複数のコマンドを蓄積可能なコマンド蓄積用バッファが設けられており、

前記マスタプロセッサから前記 DMA 実行手段および前記各プロセッサエレメントに対して、各コマンドに対する応答を待つことなく次のコマンドを発行するステップと、

発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知するステップとを具備することを特徴とするマルチプロセッサシステムの制御方法。

【請求項 7】 マスタプロセッサと、ローカルメモリを各々が有し前記マスタプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレ

メントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムの動作を制御するための制御方法であって、

前記マルチプロセッサシステムには、前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサエレメントのローカルメモリとの間のDMA転送を行うDMA実行手段、及び前記DMA実行手段および前記各プロセッサエレメントにはそれぞれ前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように複数のコマンドを蓄積可能なコマンド蓄積用バッファが設けられており、

発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知するステップと、

前記マスタプロセッサ上で並列実行される複数のスレッドそれぞれに固有の識別子を用いて、前記各コマンドをその発行元のスレッドの識別子と共に前記マスタプロセッサから前記DMA実行手段または前記各プロセッサエレメントに発行するステップと、

前記発行されたコマンドに対応する応答を該当するスレッドの識別子と共に前記DMA実行手段または前記各プロセッサエレメントから前記マスタプロセッサに発行するステップと、

前記複数のスレッドそれぞれの識別子を用いて、前記ローカルメモリの使用領域を管理テーブルによってスレッド毎に管理するステップとを具備し、

前記DMA実行手段および前記各プロセッサエレメントは、それぞれ対応するコマンド蓄積用バッファに蓄積されている前記各スレッドからのコマンドの中で、実行可能なコマンドを前記管理テーブルを参照して特定し、その特定したコマンドを実行することを特徴とするマルチプロセッサシステムの制御方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明はマルチプロセッサシステムおよびその制御方法に関し、特にグローバルメモリと各プロセッサエレメントのローカルメモリとの間でデータ／プログラ

ムの転送をDMA転送によって行うマルチプロセッサシステムおよびその制御方法に関する。

【0002】

【従来の技術】

従来より、マルチプロセッサ方式は計算機を高速化する手法として広く知られている。マルチプロセッサ方式には、メモリを共有する共有メモリ方式（密結合マルチプロセッサ方式）と、メモリをプロセッサに分散して持つ疎結合マルチプロセッサ方式とがある。

【0003】

共有メモリ方式では共有メモリを通してプロセッサ間の通信を行うことができるのでプログラミングが簡単な反面、データの整合性を保持した状態で各プロセッサが共有メモリを参照できるようにするための共有メモリアクセスに関する特別な仕組みが必要となり、ハードウェアが複雑になるという欠点がある。また、疎結合マルチプロセッサ方式では共有メモリを用いたプロセッサ間通信は行われないのでハードウェアはその分簡単化できるが、プロセッサ間通信のための機能をプロセッサ毎にそれを制御するプログラムによって提供する必要があり、プログラム作成が困難であるという問題がある。

【0004】

そこで、疎結合マルチプロセッサ方式の制御を容易に行うために、各プロセッサの制御を行うためのマスタプロセッサを設け、そのマスタプロセッサが他の各プロセッサエレメントにコマンドを送る方法が知られている。マスタプロセッサ上で他の各プロセッサエレメントに送るコマンドの順序およびタイミングを制御することで、マルチプロセッサシステム全体の動作を容易に制御することが可能となる。

【0005】

【発明が解決しようとする課題】

ところで、疎結合マルチプロセッサ方式では各プロセッサエレメントがローカルなメモリを持つが、疎結合マルチプロセッサ方式であっても各プロセッサエレメントで共通に使用可能なメモリ（グローバルメモリ）が存在した方がプログラ

ミングは容易であり、また各ローカルメモリのメモリサイズの縮小化を図ることもできる。

【0006】

しかし、一般にローカルメモリに比べ、グローバルメモリに対するアクセスにはバス調停やその他の要因で多くの時間を要する。メモリアクセスのためにプロセッサ資源が長い間専有されてしまうことはスループットの低下をもたらすことになる。

【0007】

これを改善するため、最近では、グローバルメモリと各プロセッサエレメントのローカルメモリとの間のデータ/プログラムの転送をDMAコントローラによって行う仕組みが考えられている。

【0008】

この場合、マスタプロセッサで実行されるプログラムは各プロセッサエレメントおよびDMAコントローラを制御する手順を記述したものになる。このプログラムをマルチスレッドで記述することにより、複数のプロセッサエレメントを効率よく使用することができる。

【0009】

ところが、個々のプロセッサエレメントの処理時間およびDMAにかかる時間は既知ではないので、マスタプロセッサ上で実行されるマルチスレッドプログラムで各プロセッサエレメントおよびDMAコントローラを制御する場合でも、各スレッドに対応する処理動作を該当するプロセッサエレメントに無駄無く割り当てるのは実際上困難である。空いているプロセッサエレメントの時間を減らすためには、次の二つの問題点を解決しなければならない。

【0010】

第1の問題点は、DMAとプロセッサエレメントの処理には依存性があるので、この依存性を制御するためにマスタプロセッサを用いるが、DMAおよびプロセッサエレメントの処理の終了の度にマスタプロセッサへ割り込みやスレッドの切り替えを起こしたのではマスタプロセッサの動作が非効率的であるということである。特に、マスタプロセッサによって制御すべきプロセッサエレメントの数

が増えると、マスタプロセッサへ割り込みやスレッドの切り替えの処理が頻繁に発生し、処理効率が低下することになる。

【0011】

第2の問題点は、マスタプロセッサ上で実行される2以上のスレッドに対応する処理動作をあるプロセッサエレメントに実行させる場合、あるスレッドの制御でローカルメモリにDMA転送したデータを、別のスレッドに対応する処理で使ってしまうということが起こり得る点である。

【0012】

例えば、スレッドAとスレッドBがマスタプロセッサ上で並列実行されており、それらスレッドそれぞれの制御でプロセッサエレメントがローカルメモリ上のデータを処理する場合を考える。この場合、スレッドAとスレッドBの切り換えのタイミングやDMAおよびプロセッサエレメントの処理時間との関係によっては、グローバルメモリからローカルメモリにDMA転送されたスレッドA用のデータがスレッドAに対応するプロセッサエレメントの処理動作で実際に処理される前に、スレッドAからスレッドBへの切り換えが行われ、プロセッサエレメントのスレッドBに対応する処理動作でスレッドA用のデータが使われてしまうことがある。これにより、扱うデータに不整合が生じる。

【0013】

本発明は上述の事情を考慮してなされたものであり、マスタプロセッサの負荷の増大や、扱うデータの不整合に関する問題を招くことなく、空いているプロセッサエレメントの時間を減らしてスループットの向上を図ることが可能なマルチプロセッサシステムおよびその制御方法を提供することを目的とする。

【0014】

【課題を解決するための手段】

上述の課題を解決するため、本発明は、マスタプロセッサと、ローカルメモリを各々が有し前記マスタプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレメントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムにおいて、前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサ

エレメントのローカルメモリとの間のDMA転送を行うDMA実行手段と、前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように前記DMA実行手段および前記各プロセッサエレメントにそれぞれ対応して設けられたコマンド蓄積用バッファと、発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知する未応答コマンド数管理手段とを具備することを特徴とする。

【 0 0 1 5 】

この発明によれば、DMA実行手段および各プロセッサエレメントにそれぞれコマンド蓄積用バッファが設けられているので、マスタプロセッサからDMA実行手段や各プロセッサエレメントへのコマンドは複数まとめて発行することができ、先に送ったコマンドに対する応答を待たずに次のコマンドを発行することができる。さらに、発行済みで応答が返ってきてないコマンドの数の管理が行われ、すべての発行済みのコマンドに対して応答が返ってきたときにそれがマスタプロセッサに通知される。この通知により、はじめてマスタプロセッサにスレッド切り替え等の動作をとらせれば、DMAと各プロセッサエレメントの処理の依存性を考慮しつつ、マスタプロセッサの負荷を低減することができる。よって、個々のコマンド発行およびそれに対する応答の度にスレッドの切り換えが行われるシステムに比べ、システム全体のスループットを向上させることができる。もちろん、マスタプロセッサ上のプログラムがシングルスレッドで動作している場合であっても、割り込み処理等の回数を減らすことが可能となり、マスタプロセッサの負荷を低減することができる。

【 0 0 1 6 】

また、スレッド毎に固有の識別子を設け、その識別子と共にコマンドを発行し、コマンドに対する応答も識別子と共に返すようにすることが好ましい。そして、発行済みで応答が返ってきてないコマンドの数をスレッド毎に個々に管理し、すべての発行済みのコマンドに対して応答が返ってきたスレッドに対してコマンド処理の完了を通知することにより、スレッド間の切り換えをよりスムーズに行うことが可能となる。

【0017】

また、本発明は、マスタプロセッサと、ローカルメモリを各々が有し前記マスタプロセッサからのコマンドに従ってそれぞれ制御される複数のプロセッサエレメントと、前記複数のプロセッサエレメントに共有されるグローバルメモリとを具備するマルチプロセッサシステムにおいて、前記マスタプロセッサからのコマンドに従って、前記グローバルメモリと前記各プロセッサエレメントのローカルメモリとの間のDMA転送を行うDMA実行手段と、前記マスタプロセッサが各コマンドに対する応答を待つことなく次のコマンドを発行できるように前記DMA実行手段および前記各プロセッサエレメントにそれぞれ対応して設けられたコマンド蓄積用バッファと、発行済みで応答が返ってきてないコマンドの数を管理し、すべての発行済みのコマンドに対して応答が返ってきたときにそれを前記マスタプロセッサに通知する手段と、前記マスタプロセッサ上で並列実行される複数のスレッドそれぞれに固有の識別子を用いて、前記各コマンドをその発行元のスレッドの識別子と共に前記マスタプロセッサから前記DMA実行手段または前記各プロセッサエレメントに発行する手段と、前記発行されたコマンドに対応する応答を該当するスレッドの識別子と共に前記DMA実行手段または前記各プロセッサエレメントから前記マスタプロセッサに発行する手段と、前記複数のスレッドそれぞれの識別子を用いて、前記ローカルメモリの各記憶領域毎にどのスレッドに対応するプログラムまたはデータが格納されているかを管理する管理テーブルとを具備し、前記DMA実行手段および前記各プロセッサエレメントは、それぞれ対応するコマンド蓄積用バッファに蓄積されている前記各スレッドからのコマンドの中で、実行可能なコマンドを前記管理テーブルを参照して特定し、その特定したコマンドを実行することを特徴とする。

【0018】

この構成によれば、ローカルメモリの各記憶領域毎にどのスレッドに対応するプログラムまたはデータが格納されているかを管理する管理テーブルが設けられており、上述の効果に加え、扱うデータの不整合に関する問題を解消することができる。さらに、DMA実行手段および各プロセッサエレメントは、それぞれ対応するコマンド蓄積用バッファに蓄積されている各スレッドからのコマンドの中

で、実行可能なコマンドを管理テーブルを参照して特定できるので、実行可能なコマンドから順に正しく実行することが可能となり、DMA実行手段および各プロセッサエレメントの空き時間を減らすことが可能となる。

【 0 0 1 9 】

【発明の実施の形態】

以下、図面を参照して本発明の実施形態を説明する。

【実施形態 1】

図 1 には、本発明の第 1 実施形態に係るマルチプロセッサシステムの構成が示されている。このマルチプロセッサシステムは疎結合方式のマルチプロセッサシステムであり、図示のように、バス 10、マスタプロセッサ (MP) 11、および 2 つのプロセッサエレメント (PE) 12-1, 12-2 を備えている。マスタプロセッサ (MP) 11 は 2 つのプロセッサエレメント (PE) 12-1, 12-2 を制御するためのものであり、プロセッサエレメント (PE) 12-1, 12-2 による処理動作は全てマスタプロセッサ (MP) 11 から発行されるコマンドに従って実行される。

【 0 0 2 0 】

2 つのプロセッサエレメント (PE) 12-1, 12-2 の各々はそれぞれ処理対象のプログラムおよびデータを格納するためのローカルメモリを持つ。プログラムメモリ (PLS: Program Local Storage) 13-1 およびデータローカルメモリ (DLS: Data Local Storage) 14-1 はプロセッサエレメント (PE) 12-1 のローカルメモリであり、プログラムローカルメモリ (PLS) 13-1 にはプロセッサエレメント (PE) 12-1 に実行させるべきプログラムが格納され、またデータローカルメモリ (DLS) 14-1 にはプロセッサエレメント (PE) 12-1 によって処理させるべきデータ等が格納される。同様に、プログラムメモリ (PLS: Program Local Storage) 13-2 およびデータローカルメモリ (DLS: Data Local Storage) 14-2 はプロセッサエレメント (PE) 12-2 のローカルメモリであり、プログラムローカルメモリ (PLS) 13-2 にはプロセッサエレメント (PE) 12-2 に実行させるべき

プログラムが格納され、またデータローカルメモリ (DLS) 14-2にはプロセッサエレメント (PE) 12-2によって処理させるべきデータ等が格納される。

【0021】

さらに、バス10にはDMAコントローラ15-1, 15-2が接続されると共に、メモリコントローラ16を介してグローバルメモリ (GM) 17が接続されている。グローバルメモリ (GM) 17はプロセッサエレメント (PE) 12-1, 12-2が共通に使用可能な一種の共有メモリであり、ここにはマスタプロセッサ (MP) 11によって実行されるプログラムおよび処理データが格納されるほか、プロセッサエレメント (PE) 12-1, 12-2によって実行されるプログラムおよび処理データも格納される。このグローバルメモリ (GM) 17から必要なプログラムをプログラムローカルメモリ (PLS) 13-1, 13-2にそれぞれ転送することにより、プログラムローカルメモリ (PLS) 13-1, 13-2のメモリサイズを大きくせずに、比較的大きなプログラムでもそれをプロセッサエレメント (PE) 12-1, 12-2にそれぞれ実行させることができる。

【0022】

また、グローバルメモリ (GM) 17とデータローカルメモリ (DLS) 14-1, 14-2それぞれとの間のデータ転送により、処理対象のデータをグローバルメモリ (GM) 17からデータローカルメモリ (DLS) 14-1, 14-2に転送してプロセッサエレメント (PE) 12-1, 12-2にそれぞれ処理させたり、それぞれの処理結果のデータをデータローカルメモリ (DLS) 14-1, 14-2からグローバルメモリ (GM) 17に転送するなどの制御を容易に行うことができる。

【0023】

DMAコントローラ15-1, 15-2は、それぞれマスタプロセッサ (MP) 11からのコマンドに従って、対応するローカルメモリとグローバルメモリ (GM) 17との間でデータ/プログラムのDMA転送を実行する。これらDMAコントローラ15-1, 15-2の存在により、プロセッサエレメント12-1

、 1 2 - 2 およびマスタプロセッサ (MP) 1 1 の資源を専有することなく、ローカルメモリとグローバルメモリ (GM) 1 7 との間のデータ/プログラムの転送を効率よく行うことができる。

【 0 0 2 4 】

上述したように、プロセッサエレメント 1 2 - 1、 1 2 - 2 の処理および DMA コントローラ 1 5 - 1、 1 5 - 2 の動作は全てマスタプロセッサ (MP) 1 1 によって制御される。この制御処理は、マスタプロセッサ (MP) 1 1 上で実行されるマルチスレッドプログラムによって実行される。マルチスレッドプログラムを構成する複数のスレッドはマスタプロセッサ (MP) 1 1 上で並列に実行され、それらスレッド毎に必要な演算処理等がプロセッサエレメント 1 2 - 1 または 1 2 - 2 を用いて実行される。つまり、各スレッドは DMA コントローラ 1 5 - 1 または 1 5 - 2、 およびプロセッサエレメント 1 2 - 1 または 1 2 - 2 に送るコマンドの順序およびタイミングを制御することで、必要な演算処理をプロセッサエレメントに実行させ、その処理結果をグローバルメモリ (GM) 1 7 上に転送させることができる。基本的には、以下の手順で処理が行われる。

【 0 0 2 5 】

- 1) グローバルメモリ (GM) からプログラムローカルメモリ (PLS) へのプログラムの DMA 転送
- 2) グローバルメモリ (GM) からデータローカルメモリ (DLS) への処理対象データの DMA 転送
- 3) プロセッサエレメントによる処理
- 4) データローカルメモリ (DLS) からグローバルメモリ (GM) への処理結果データの転送

本マルチプロセッサシステムは例えば産業用または家庭用機器への組み込みコンピュータとして使用される。この場合、図 1 に示した複数の機能モジュールのうち、グローバルメモリ (GM) 1 7 を除く他の全てのモジュールは 1 チップ LSI 上に集積形成することができる。

【 0 0 2 6 】

さらに、本マルチプロセッサシステムでは、一連の制御に必要な幾つかのコマ

ンドをまとめて扱えるようにするために、プロセッサエレメント (PE) 12-1, 12-2 にはそれぞれ複数のコマンドを蓄積可能なコマンドプーリングバッファ 121 が設けられ、また DMA コントローラ 15-1, 15-2 にもそれぞれ複数のコマンドを蓄積可能なコマンドプーリングバッファ 131 が設けられている。これらコマンドプーリングバッファ 121, 131 の各々は追い越し制御可能な一種のコマンドキュー (プール) である。

【 0 0 2 7 】

これらコマンドプーリングバッファ 121, 131 の存在により、マスタプロセッサ (MP) 11 から DMA コントローラ 15-1, 15-2 や各プロセッサエレメント (PE) へのコマンドは複数まとめて発行することができ、先に送ったコマンドに対する応答を待たずに次のコマンドを発行することができる。さらに、発行済みで応答が返ってきてないコマンドの数の管理はマスタプロセッサ (MP) 11 に設けられたカウンタアレイ 111 によって行われ、すべての発行済みのコマンドに対して応答が返ってきたときにそのことがマスタプロセッサ (MP) 11 に通知される。この通知により、はじめてマスタプロセッサ (MP) 11 にスレッド切り替え等の動作をとらせることにより、DMA と各プロセッサエレメント 12-1, 12-2 の処理の依存性を考慮しつつ、マスタプロセッサ (MP) 11 の負荷を低減することができる。この場合、実際には、スレッド間の切り換えをよりスムーズに行うために、発行済みで応答が返ってきてないコマンドの数の管理はスレッド毎に行うことが好ましい。

【 0 0 2 8 】

すなわち、本実施形態では、スレッドごとに固有の識別子 VTID (Virtual Thread ID) を付ける。さらに、カウンタアレイ 111 には VTID ごとに固有のカウンタを設け、マスタプロセッサ (MP) 11 からのコマンドの発行で該当するカウンタをインクリメントする。DMA コントローラ 15-1, 15-2 やプロセッサエレメント (PE) 12-1, 12-2 へはコマンドと共に VTID を送出するようにし、DMA コントローラ 15-1, 15-2 やプロセッサエレメント (PE) 12-1, 12-2 からの応答には対応した VTID を共に返すようにする。マスタプロセッサ (MP) 11 では応答と共に返

ってきた V T I D に対応したカウンタをデクリメントする。カウンタが 0 になれば、はじめて該当するスレッドにそれを通知してスリープ状態から実行状態に起こすことにより、実行スレッドの切り換えをよりスムーズに行うことが可能となる。図 2 にこの様子を示す。

【 0 0 2 9 】

図 2 (a) は従来の場合に相当するものであり、マスタプロセッサ (M P) 1 1 上で実行されているスレッドはプロセッサエレメント (P E) へ 2 回コマンドを発行したことで 2 回の応答を受けている。この例ではコマンドを発行したときにスレッドをスリープさせ、応答があったときコマンドを再び起こしているが、これは割り込みであったり、スリープの代わりにビジーウェイトであっても構わない。

【 0 0 3 0 】

図 2 (b) は本実施形態の場合であり、マスタプロセッサ (M P) 1 1 上で実行されているスレッドからプロセッサエレメント (P E) に最初にコマンドを上げた時点でカウンタの値が 1 になる。その後、 2 個目のコマンドをプロセッサエレメント (P E) に投げてこのスレッドをスリープさせる。この時、カウンタの値は 2 になっている。その後、プロセッサエレメント (P E) より最初のコマンドに対する応答が返ってきてカウンタの値は 1 になる。最後にプロセッサエレメント (P E) より 2 個目のコマンドに対応する応答が返ってきてカウンタの値が 0 になり、スリープしていたスレッドを起こす。

【 0 0 3 1 】

あるスレッドがスリープしている間は他のスレッドがマスタプロセッサ (M P) 1 1 上で実行されるので、一連の処理に必要な幾つかのコマンド処理待ちの間にマスタプロセッサ (M P) 1 1 は他の処理を行うことができる。コマンドをいくつ発生してからスリープするかについては各スレッドの記述によって決まる。

【 0 0 3 2 】

以上のようにカウンタという簡単なハードウェアを設けることにより、マスタプロセッサ (M P) 1 1 上のスレッドをスリープさせたり再び起こしたりするオーバーヘッドが削減される。もちろん、割り込みや、ビジーウェイトなどを使っ

た場合も同様にオーバーヘッドは削減される。

【 0 0 3 3 】

このような仕組みを実現するためのハードウェアは以下の図 3 のような構成が考えられる。

【 0 0 3 4 】

図 3 ではマスタプロセッサ (MP) 1 1 とそれにつながるバス 1 0 を示している。マスタプロセッサ (MP) 1 1 は図示のようにプロセッサモジュール 2 0 1 、バスコントローラ 2 0 2 、およびカウンタアレイ 2 0 3 を備えている。このカウンタアレイ 2 0 3 は図 1 のカウンタアレイ 1 1 1 と同じものである。

【 0 0 3 5 】

プロセッサモジュール 2 0 1 からコマンドを発行するとき、コマンドは発行元のスレッドの VTID と共にバスコントローラ 2 0 2 に送られる。また、プロセッサモジュール 2 0 1 からカウンタアレイ 2 0 3 には VTID が与えられる。カウンタアレイ 2 0 3 では VTID に対応したカウンタをインクリメントする。一方、バス 1 0 を介して VTID とともに応答が返ってくると、カウンタアレイ 2 0 3 の対応するカウンタをデクリメントする。カウンタが 0 になった場合はプロセッサモジュール 2 0 1 への通知が行われ、該当する VTID のスレッドが起こされ、すべての発行済みのコマンドに対して応答が返ってきたスレッドに対してコマンド処理の完了が通知される。

【 0 0 3 6 】

次に、図 4 および図 5 のフローチャートを参照して、マスタプロセッサ (MP) 1 1 上で実行される各スレッドの動作と、各スレッドからのコマンドで制御される DMA コントローラおよびプロセッサエレメントの動作について説明する。

【 0 0 3 7 】

図 4 は、マスタプロセッサ (MP) 1 1 上で実行される各スレッドの動作を示している。まず、スレッドは DMA コントローラ 1 5 - 1, 1 5 - 2 やプロセッサエレメント (PE) 1 2 - 1, 1 2 - 2 に対して、VTID とコマンドの組を N 個順次発行する (ステップ S 1 0 1)。この後、スレッドは、対応する VTID のカウンタを N にセットした後 (ステップ S 1 0 2)、スリープ状態に移行さ

れる（ステップ S 1 0 3）。そして、カウンタアレイ 2 0 3 から通知が来ると、カウンタ値が 0 になったカウンタに対応する V T I D のスレッドがスリープ状態から起こされ（ステップ S 1 0 4）、その起こされたスレッドによる処理が続行される。

【 0 0 3 8 】

図 5 は、DMA コントローラ 1 5 - 1，1 5 - 2 およびプロセッサエレメント（P E）1 2 - 1，1 2 - 2 の動作を示している。

【 0 0 3 9 】

DMA コントローラまたはプロセッサエレメントでマスタプロセッサ（M P）1 1 からのコマンドが受け付けられると（ステップ S 1 1 1）、まずそのコマンドに対応する V T I D がセーブされた後（ステップ S 1 1 2）、コマンド処理が実行される（ステップ S 1 1 3）。このコマンド処理では、DMA コントローラ 1 5 - 1，1 5 - 2 の場合にはコマンドで指定された DMA 転送が実行され、プロセッサエレメントの場合にはコマンドで指定されたプログラム実行処理が行われることになる。

【 0 0 4 0 】

コマンド処理が終了すると、そのコマンドを実行した DMA コントローラまたはプロセッサエレメントは、そのコマンドに対する応答として、セーブしておいた V T I D と共にコマンド完了通知をマスタプロセッサ（M P）1 1 に発行する（ステップ S 1 1 4）。

【 0 0 4 1 】

以上のように、本マルチプロセッサシステムによれば、スレッドごとの識別子 V T I D とマスタプロセッサ（M P）1 1 に設けた V T I D ごとのカウンタによって発行済みで応答のないコマンド数をスレッドごとに管理し、発行済みのコマンド処理がすべて終わったときにマスタプロセッサ（M P）1 1 にそれを通知してスレッド間切り換えなどのトリガとなるインタラクションを起こすことで、マスタプロセッサ（M P）1 1 を効果的に使用することができる。

【 0 0 4 2 】

なお、本実施形態の仕組みを用いることにより、マスタプロセッサ（M P）1

1 上のプログラムがシングルスレッドで動作している場合であっても割り込み処理等の回数を減らすことが可能となり、マスタプロセッサ (MP) 1 1 の負荷を低減することができる。

【 0 0 4 3 】

〔実施形態 2〕

次に、本発明の第 2 実施形態として、DMA 及び各 PE の制御に関する具体的な手順およびローカルメモリ管理のための仕組みについて説明する。

図 6 には、本第 2 実施形態のマルチプロセッサシステムの構成が示されている。本マルチプロセッサシステムでは、ローカルメモリを管理するためのテーブルを設け、どの VTID に対応したプログラムやデータがローカルメモリに入っているかを管理する。これにより上述の第 1 実施形態の効果に加え、扱うデータの不整合に関する問題を解消することができる。また、DMA コントローラやプロセッサエレメントは、対応するローカルメモリの管理テーブルを参照することにより、コマンドのプール中より実行可能なものを特定できるので、実行可能なコマンドから順に正しく実行することが可能となる。以下、具体的に説明する。

【 0 0 4 4 】

図 6 に示されているように、DMA コントローラ 1 5 - 1 には、プロセッサエレメント (PE) 1 2 - 1 のローカルメモリを管理する管理テーブルとして PLS 管理テーブル 3 0 1 と DLS 管理テーブル 3 0 2 が設けられている。

【 0 0 4 5 】

PLS 管理テーブル 3 0 1 はプログラムローカルメモリ (PLS) 1 3 - 1 の記憶領域毎にどのスレッドに対応するプログラムが格納されているかを管理し、また DLS 管理テーブル 3 0 2 はデータローカルメモリ (DLS) 1 4 - 1 の記憶領域毎にどのスレッドに対応するデータが格納されているかを管理する。

【 0 0 4 6 】

また、DMA コントローラ 1 5 - 1 に設けられた PLS コマンドテーブル 3 0 3 および DLS コマンドテーブル 3 0 4 は図 1 のコマンドプーリングバッファ 1 5 1 に相当するものであり、PLS コマンドテーブル 3 0 3 にはプログラムローカルメモリ (PLS) 1 3 - 1 とグローバルメモリ (GM) 1 7 との間の DMA

転送に関するコマンドが蓄積され、またDLSコマンドテーブル304にはデータローカルメモリ(DLS)14-1とグローバルメモリ(GM)17との間のDMA転送に関するコマンドが蓄積される。

【0047】

プロセッサエレメント(PE)12-1に設けられたPEコマンドテーブル305は、図1のコマンドプーリングバッファ121に相当し、ここにはプロセッサエレメント(PE)12-1に対するコマンドが蓄積される。

【0048】

同様に、DMAコントローラ15-2にも、プロセッサエレメント(PE)12-2のローカルメモリを管理する管理テーブルとしてPLS管理テーブル401とDLS管理テーブル402が設けられている。PLS管理テーブル401はプログラムローカルメモリ(PLS)13-2の記憶領域毎にどのスレッドに対応するプログラムが格納されているかを管理し、またDLS管理テーブル402はデータローカルメモリ(DLS)14-2の記憶領域毎にどのスレッドに対応するデータが格納されているかを管理する。また、DMAコントローラ15-2に設けられたPLSコマンドテーブル403およびDLSコマンドテーブル404は図1のコマンドプーリングバッファ151に相当するものであり、PLSコマンドテーブル403にはプログラムローカルメモリ(PLS)13-2とグローバルメモリ(GM)17との間のDMA転送に関するコマンドが蓄積され、またDLSコマンドテーブル404にはデータローカルメモリ(DLS)14-2とグローバルメモリ(GM)17との間のDMA転送に関するコマンドが蓄積される。プロセッサエレメント(PE)12-2に設けられたPEコマンドテーブル405は、図1のコマンドプーリングバッファ121に相当し、ここにはプロセッサエレメント(PE)12-2に対するコマンドが蓄積される。

【0049】

なお、DMAコントローラは必ずしも2つ設ける必要はなく、図7に示すように一個のDMAコントローラ15のみをバス10に接続し、そのDMAコントローラ15によってプロセッサエレメント(PE)12-1、12-2それぞれのローカルメモリとグローバルメモリ(GM)17との間のDMA転送を行うよう

にしても良い。これは第1実施形態についても同様である。

【0050】

また、一個のDMAコントローラ15のみを使用する場合には、プロセッサエレメント (PE) 12-1 側に対応するPLS管理テーブル301、DLS管理テーブル302、PLSコマンドテーブル303、DLSコマンドテーブル304と、プロセッサエレメント (PE) 12-2 側に対応するPLS管理テーブル401、DLS管理テーブル402、PLSコマンドテーブル403、DLSコマンドテーブル404が全てDMAコントローラ15に設けられることになる。また、PEコマンドテーブル305、405の実際の実装位置についても必ずしもプロセッサエレメント (PE) 12-1, 12-2にする必要はなく、DMAコントローラ15に設けても良い。第1実施形態についても同様である。

【0051】

次に、各テーブルの具体的な構成について説明する。

本第2実施形態では、複数のスレッドが同じプロセッサエレメント (PE) およびそのローカルメモリPLS, DLSを使用しながら並列動作することを前提としている。その為に、プロセッサエレメント (PE) 12-1 のプログラムローカルメモリ (PLS) 13-1 およびデータローカルメモリ (DLS) 14-1 についてはそれぞれPLS管理テーブル301およびDLS管理テーブル302によってその使用領域をスレッド毎に管理し、またプロセッサエレメント (PE) 12-2 のプログラムローカルメモリ (PLS) 13-2 およびデータローカルメモリ (DLS) 14-2 についてはそれぞれPLS管理テーブル401およびDLS管理テーブル402によってその使用領域をスレッド毎に管理している。

【0052】

これらテーブルの構造はプロセッサエレメント (PE) 12-1, 12-2 のどちらについても同じであるので、以下ではプロセッサエレメント (PE) 12-1 および12-2について共通に説明することにする。

【0053】

PLS管理テーブルとDLS管理テーブルはハードウェア簡単化のため、ペー

ジ単位で管理される。例えば、ページサイズが4 K b y t eでP L S, D L Sのサイズがそれぞれ6 4 K b y t eであれば、それぞれ1 6ページとして管理されることになる。この場合、P L S管理テーブルはページに対応する1 6のエントリを持ち、それぞれのエントリがそのページに入っているプログラムのV T I Dを示すことになる。図8にP L S管理テーブルの例を示す。

【 0 0 5 4 】

図8では、ページ1, 2, 3にV T I D 4 4のプログラムが入っていてその他のページは使われていない様子が示されている。

【 0 0 5 5 】

次に各プロセッサエレメント（P E）で処理されるデータまたは処理された結果のデータが入るD L Sを管理するためのD L S管理テーブルについて説明する。図9はD L S管理テーブルの例である。

【 0 0 5 6 】

図9（a）ではページ0, 1はV T I D 4 4のデータが入っている（Dは処理されるべきデータが入っていることを示す）。またページ2, 3はV T I D 4 4のために領域が予約されていることを示している（Rは予約されている領域であることを示す）。領域の予約は、対応するプロセッサエレメント（P E）がこの領域を処理結果データの書き込みのために使用することを示している。予約されたページはプロセッサエレメント（P E）がデータを書き込むとさらにDの部分にビットが立ち図9（b）のようになる。

【 0 0 5 7 】

次に、P L Sコマンドテーブルの構成を図10に示す。P L Sコマンドテーブルはグローバルメモリ（G M）1 7よりプログラムローカルメモリ（P L S）に転送するDMAのコマンドを入れておくテーブルである。マスタプロセッサ（M P）1 1は、プロセッサエレメント（P E）に実行させるプログラムをグローバルメモリ（G M）1 7より制御対象のプロセッサエレメント（P E）のプログラムローカルメモリ（P L S）に転送するためにこのテーブルにDMAのコマンドを登録する。

【 0 0 5 8 】

図10では、PLSコマンドテーブルに二つのエントリが存在している、一つはVTID44でグローバルメモリ(GM)17のアドレス0×120000からプログラムローカルメモリ(PLS)のページ0, 1, 2に転送するコマンド、もう一つは、VTID50でグローバルメモリ(GM)17のアドレス0×140000からプログラムローカルメモリ(PLS)のページ14, 15に転送するコマンドである。PLSコマンドテーブルの内容はPLS管理テーブルと比較され、転送先ページが空いている場合にはそのエントリに入っているコマンドのDMAが実行され、DMAが実行されるとそのエントリは消去される。複数のエントリがDMA実行可能であるときは先入れ先出し方式で実行される。エントリの数にはハードウェア的な制約があるのでエントリがいっぱいときはマスタプロセッサ(MP)11のコマンド発行元のスレッドがスリープするか、エントリが空くまで待つなどの処理を行う。

【0059】

PLSコマンドに対するDMAコントローラの処理手順を図11に示す。DMAコントローラは、まず、PLSコマンドテーブルからコマンドを取得し(ステップS121)、そのコマンドで指定される転送先ページがプログラムローカルメモリ(PLS)上で空いているか否かをPLS管理テーブルを参照して判断する(ステップS122)。空いていない場合には、次のコマンドエントリに対する処理が行われる。一方、空いている場合には、DMAコントローラはその取得したコマンドの処理、つまりグローバルメモリ(GM)17からプログラムローカルメモリ(PLS)へのプログラムのDMA転送、を実行する(ステップS123)。次いで、DMAコントローラは、PLS管理テーブルの該当エントリにDのフラグをセットしてその内容を更新し(ステップS124)、そしてPLSコマンドテーブルから該当するコマンドエントリの内容を削除した後に(ステップS125)、実行したコマンドに対応するVTIDと共にコマンド完了通知をマスタプロセッサ(MP)11に発行する(ステップS126)。

【0060】

このようにして、DMAコントローラは、PLSコマンドテーブルに蓄積されている複数のコマンドを、実行可能なものから順次実行する。

【0061】

次にDLSコマンドテーブルの構成を図12に示す。DLSコマンドテーブルはグローバルメモリ（GM）17よりデータローカルメモリ（DLS）にデータを転送するDMAのコマンドおよびデータローカルメモリ（DLS）からグローバルメモリ（GM）17にデータを転送するコマンドを、マスタプロセッサ（MP）11より受け取っていれておくためのテーブルである。

【0062】

図12ではDLSコマンドテーブルに二つのエントリがある。一つはVTID44でグローバルメモリ（GM）17のアドレス0×20000からデータローカルメモリ（DLS）へ転送することを示している。方向フィールドはDMA転送の方向を示す。「D」がグローバルメモリ（GM）17からデータローカルメモリ（DLS）への転送、「G」がデータローカルメモリ（DLS）よりグローバルメモリ（GM）17への転送を示している。また、転送ページのフィールドにおける「D」はデータが転送される領域を、「R」はプロセッサエレメント（PE）による処理結果データの書き込みに使用すべき予約領域を示している。例ではページ0，1にGMよりデータが転送され、ページ2，3は書き込みのために予約されている。

【0063】

二つ目のエントリにはPEにより書き込まれた処理結果データをグローバルメモリ（GM）17へ転送するためのコマンドが入っている。同じVTID44のスレッドで扱われるのでVTIDは44である。グローバルメモリ（GM）17のアドレスは0×40000で今度は転送の方向は「G」となり、データローカルメモリ（DLS）よりグローバルメモリ（GM）17への転送を示している。転送ページのフィールドではページ2，3が転送されることを示している。

【0064】

PLSコマンドテーブルと同様に、エントリの数にはハードウェア的な制約があるのでエントリがいっぱいときはマルチプロセッサ（MP）のコマンド発行元のスレッドがスリープするか、エントリが空くまで待つなどの処理を行う。DLSコマンドテーブルからDMAコマンドが実行されるのは、次のような条件に

なる。

【 0 0 6 5 】

(1) 方向フィールドがDの時： 転送ページで示すDおよびRがDLS管理テーブルで空いており、かつPLS管理テーブルに同じVTIDが存在する時。つまり、DMAを実行した後、プロセッサエレメント（PE）が処理するプログラムがすでにプログラムローカルメモリ（PLS）に入っていて、しかもデータを入れたり出したりするデータローカルメモリ（DLS）の領域が他のスレッドとぶつからないことを保障している。

【 0 0 6 6 】

(2) 方向フィールドがGのとき： 転送ページで示すDにDLS管理テーブルのDにビットがたっており、またVTIDが一致していること。すなわちVTIDで示すスレッドのプロセッサエレメント（PE）の処理が終わってデータがすでにデータローカルメモリ（DLS）に書き込まれてグローバルメモリ（GM）17に転送すべきデータが揃ったことを保障している。

【 0 0 6 7 】

DLSコマンドテーブルは同じVTIDに対しては先入れ先出しで処理されるが、違うVTIDでの順序は保障する必要はない。

【 0 0 6 8 】

DLSコマンドに対するDMAコントローラの処理手順を図13に示す。DMAコントローラは、まず、DLSコマンドテーブルからコマンドを取得し（ステップS131）、そのコマンドで指定される方向フィールドをチェックする（ステップS132）。方向フィールドがD、つまりデータローカルメモリ（DLS）への転送を示すコマンドであれば、DMAコントローラは、その取得したコマンドで指定されるD、Rの転送先ページがデータローカルメモリ（DLS）上で空いているか否かをDLS管理テーブルを参照して判断する（ステップS133）。空いている場合には、さらに、取得したコマンドのVTIDと同じVTIDを持つプログラムがプログラムローカルメモリ（PLS）に既に存在しているか否かをPLS管理テーブルを参照して判断する（ステップS134）。同じVTIDを持つプログラムがプログラムローカルメモリ（PLS）に存在している場

合には、上述の（１）の条件が満たされているので、DMAコントローラは取得したコマンドの処理、つまりグローバルメモリ（GM）１７からデータローカルメモリ（DLS）へのデータのDMA転送、を実行する（ステップS１３５）。次いで、DMAコントローラは、DLS管理テーブルの該当エントリにD、Rのフラグをセットしてその内容を更新し（ステップS１３６）、そしてDLSコマンドテーブルから該当するコマンドエントリの内容を削除した後に（ステップS１３７）、実行したコマンドに対応するVTIDと共にコマンド完了通知をマスタプロセッサ（MP）１１に発行する（ステップS１３８）。

【 0 0 6 9 】

上述の（１）の条件が満たされていない場合、つまり、取得したコマンドで指定されるD、Rの転送先ページが空いていない場合や、空いていても同じVTIDのプログラムが存在しない場合には、コマンド処理は行われず、次のコマンドエントリに対する処理が行われる。

【 0 0 7 0 】

一方、取得したコマンドで指定される方向フィールドがG、つまりグローバルメモリ（GM）１７への転送を示すコマンドであれば、DMAコントローラは、その取得したコマンドのDの転送ページで指定されるデータローカルメモリ（DLS）上の位置にデータが既に存在するか否かをDLS管理テーブルを参照して判断する（ステップS１３９）。存在する場合には、そのデータに対応するVTIDTが取得したコマンドのVTIDTと一致しているかどうか判断される（ステップS１４０）。VTIDが一致する場合には、上述の（２）の条件が満たされているので、DMAコントローラは取得したコマンドの処理、つまりデータローカルメモリ（DLS）からグローバルメモリ（GM）１７へのデータのDMA転送、を実行する（ステップS１４１）。次いで、DMAコントローラは、DLS管理テーブルの該当エントリのDフラグをリセットしてその内容を更新し（ステップS１４２）、そしてDLSコマンドテーブルから該当するコマンドエントリの内容を削除した後に（ステップS１４３）、実行したコマンドに対応するVTIDと共にコマンド完了通知をマスタプロセッサ（MP）１１に発行する（ステップS１４４）。

【0071】

上述の(2)の条件が満たされていない場合、つまり、取得したコマンドで指定されるDの転送ページが存在しない場合や、存在してもVTIDが異なる場合には、コマンド処理は行われず、次のコマンドエントリに対する処理が行われる。

【0072】

このようにして、DMAコントローラは、DLSコマンドテーブルに蓄積されている複数のコマンドを、実行可能なものから順次実行する。

【0073】

以上で述べたような構造は、DMAコントローラ内にあり、ステートマシンで管理更新されている。

【0074】

次にPEコマンドテーブルの構成を図14に示す。PEコマンドテーブルはマスタプロセッサ(MP)11からプロセッサエレメント(PE)へのコマンドを入れておくためのテーブルである。

【0075】

図14では1番目のエントリがVTID44で開始ページは0。これはプロセッサエレメント(PE)がプログラムの実行を始めるべきプログラムローカルメモリ(PLS)上のページがかかっている。使用ページフィールドはプログラムローカルメモリ(PLS)上のページ毎にデータローカルメモリ(DLS)上のどのページのデータを使うかが示されている。例では0と1ページのデータを使うことが示されている。PEコマンドテーブル内のコマンドが実際に実行されるのはPEコマンドテーブルにある使用ページがDLS管理テーブルのDで示されているページと一致し、且つVTIDが一致する場合である。すなわち、プロセッサエレメント(PE)で実行するデータがデータローカルメモリ(DLS)に揃っていることを意味する。データローカルメモリ(DLS)にプロセッサエレメント(PE)で使用するデータが揃っているということはすでにプログラムローカルメモリ(PLS)には対応するプログラムが用意されているということなのですぐに実行が始められる。プロセッサエレメント(PE)で実行が終了した

らPEコマンドテーブルから対応するコマンドを消し、DLS管理テーブルを更新する(RのところにDも立てる)。また、このVTIDではプログラムローカルメモリ(PLS)に入れたプログラムを必要としないのであれば、対応するPLS管理テーブルも更新する。必要とするかどうかはプログラム中で明示的に指定するものとする。

【0076】

PEコマンドテーブルは同じVTIDに対しては先入れ先出し方式で処理されるが、違うVTIDでの順序は保障する必要はない。DLSコマンドテーブルは同じVTIDに対しては先入れ先出し方式で処理されるが、違うVTIDでの順序は保障する必要はない。

【0077】

PEコマンドに対するプロセッサエレメント(PE)の処理手順を図15に示す。プロセッサエレメント(PE)は、まず、PEコマンドテーブルからコマンドを取得し(ステップS151)、そのコマンドで指定される使用ページがデータローカルメモリ(DLS)に存在しているか否かをDLS管理テーブルを参照して判断する(ステップS152)。存在する場合には、そのデータのVTIDが取得したコマンドのVTIDに一致するかどうか判断される(ステップS153)。一致するならば、プロセッサエレメント(PE)は、取得したコマンドで指定される処理、つまりプログラムローカルメモリ(PLS)のプログラムの実行処理を行う(ステップS154)。このプログラム実行により処理されたデータは該当するデータローカルメモリ(DLS)上の位置に格納され、DLS管理テーブルの更新が行われる(RのところにDも立てる)。そして、実行したコマンドに対応するVTIDと共にコマンド完了通知をマスタプロセッサ(MP)11に発行する(ステップS155)。

【0078】

このようにして、プロセッサエレメント(PE)は、PEコマンドテーブルに蓄積されている複数のコマンドを、実行可能なものから順次実行する。

【0079】

図16には、DMAコントローラおよびプロセッサエレメント(PE)と各テ

ーブルとの関係が示されている。ここでは、プロセッサエレメント (PE) 12-1 側に着目して説明するが、プロセッサエレメント (PE) 12-2 側についても同様である。

【0080】

図16で示すように、マスタプロセッサ (MP) よりPLSコマンドテーブル303、DLSコマンドテーブル304、PEコマンドテーブル305にコマンドが登録され、DMAコントローラ15-1およびプロセッサエレメント (PE) 12-1によるコマンドの実行に応じてPLS管理テーブル301およびDLS管理テーブル302が更新される。PLS管理テーブル301およびDLS管理テーブル302がローカルメモリの管理を適切に行っているため、実行中のデータが他のスレッドにより書き換えられたりすることはないように制御される。

【0081】

また、プロセッサエレメント (PE) 12-1に関しては、処理すべきデータが存在する時だけ対応するプログラムが動作するようになっている。すなわち、本発明によってDMAとプロセッサエレメント (PE) の処理の依存関係が適切に表現されており、マスタプロセッサ (MP) の介在無しに、DMAとプロセッサエレメント (PE) を正しい順序関係で動作させることができ、マスタプロセッサ (MP) への負荷を減らすことができる。さらに、複数のスレッドが資源を共有することによって、例えばDMA転送によって処理データをローカルメモリに用意している間にプロセッサエレメント (PE) を他のスレッドによるプログラム処理で使うなどの効率的な動作が可能になる。この方法は、ローカルメモリをダブルバッファの構造にし、一つのスレッドに片方のバッファの処理をもう一つのスレッドにもう一方のバッファの処理を行わせることで特に有効になる。

【0082】

図17には、本マルチプロセッサシステム全体の動作の様子が模式的に示されている。

【0083】

マスタプロセッサ (MP) 上で実行されるあるスレッドは、まず、そのスレッドのVTID (ここではVTID=1) と一緒にプロセッサエレメントPE#1

に対応するPEテーブルにコマンド（PEコマンド）を登録する。次いで、そのスレッドは、プロセッサエレメントPE # 1に実行させるべきプログラムをデータローカルメモリ（DLS）に全て転送するのに必要な幾つかのコマンド（PLSコマンド）をVTID（ここではVTID=1）と一緒にPLSコマンドテーブルに登録する。この後、そのスレッドは、プロセッサエレメントPE # 1に処理させるべきデータをデータローカルメモリ（DLS）に全て転送するのに必要な幾つかのコマンド（DLSコマンド 方向D）をVTID（ここではVTID=1）と一緒にDLSコマンドテーブルに登録し、また処理結果データをグローバルメモリ（GM）に全て転送させるために必要な幾つかのコマンド（DLSコマンド 方向G）をVTID（ここではVTID=1）と一緒にDLSコマンドテーブルに登録する。これで、スレッドはコマンド処理待ちとなり、スリープ状態に移行し、他のスレッドがマスタプロセッサ（MP）上で実行される。もちろん、一連のコマンド登録が途中でできなくなった場合には、その時点でスレッドの切り換えを行っても良い。

【 0 0 8 4 】

DMAコントローラ # 1によりプログラムローカルメモリ（PLS）へのプログラムのDMA転送、およびデータローカルメモリ（DLS）への処理対象データのDMA転送が行われると、プロセッサエレメントPE # 1によるPEコマンドの処理が可能となり、PLS上のプログラムがプロセッサエレメントPE # 1によって実行され、処理結果データがDLSに書き込まれる。これにより、DMAコントローラ # 1によるコマンド（DLSコマンド 方向G）の処理が可能となり、処理結果データがグローバルメモリにDMA転送される。この時点で、VTID=1に関する未応答のコマンドが無くなり、VTID=1のスレッドが再び起こされる。

【 0 0 8 5 】

以上のように、本プロセッサシステムによれば、ローカルメモリの各記憶領域毎にどのスレッドに対応するプログラムまたはデータが格納されているかを管理することにより、扱うデータの不整合に関する問題を解消することができる。さらに、DMAコントローラおよび各プロセッサエレメント（PE）は、それぞれ

対応するコマンド蓄積用バッファに蓄積されている各スレッドからのコマンドの中で、実行可能なコマンドを特定できるので、実行可能なコマンドから順に正しく実行することが可能となり、DMAコントローラおよび各プロセッサエレメント（PE）の空き時間を減らすことが可能となる。

【0086】

なお、ここでは本システムを第1実施形態と第2実施形態に分けて説明したが、各実施形態の機能は適宜組み合わせることができる。また、マスタプロセッサ（MP）によって実行されるプログラムはマルチスレッド型のものであることが最も好ましいが、並列実行可能な複数のプログラム実行単位の各々が、必要な演算処理などをプロセッサエレメントを用いて実行する構成であればよい。さらに、スレッド切り換えなどの機能はマスタプロセッサ（MP）上で実行されるオペレーティングシステムのカーネルに組み込んだり、或いは専用のプログラムを用いてスレッド切り換えの機能を実現してもよい。

【0087】

また、本発明は、上記各実施形態に限定されるものではなく、実施段階ではその要旨を逸脱しない範囲で種々に変形することが可能である。更に、上記実施形態には種々の段階の発明が含まれており、開示される複数の構成要件における適宜な組み合わせにより種々の発明が抽出され得る。例えば、実施形態に示される全構成要件から幾つかの構成要件が削除されても、発明が解決しようとする課題の欄で述べた課題の少なくとも1つが解決でき、発明の効果の欄で述べられている効果が得られる場合には、この構成要件が削除された構成が発明として抽出され得る。

【0088】

【発明の効果】

以上詳述した如く本発明によれば、マスタプロセッサの負荷の増大や、扱うデータの不整合に関する問題を招くことなく、空いているプロセッサエレメントの時間を減らしてスループットの向上を図ることが可能となる。

【図面の簡単な説明】

【図1】

本発明の第 1 実施形態に係るマルチプロセッサシステムの構成を示すブロック図。

【図 2】

同実施形態における未応答コマンド数に応じたスレッドの状態遷移制御を説明するための図。

【図 3】

同実施形態におけるマスタプロセッサの構成を示すブロック図。

【図 4】

同実施形態におけるマスタプロセッサ上で実行されるスレッドの動作を示すフローチャート。

【図 5】

同実施形態における DMA コントローラおよびプロセッサエレメントの動作を示すフローチャート。

【図 6】

本発明の第 2 実施形態に係るマルチプロセッサシステムの構成を示すブロック図。

【図 7】

同実施形態のマルチプロセッサシステムの他の構成例を示すブロック図。

【図 8】

同実施形態における P L S 管理テーブルの構成を示す図。

【図 9】

同実施形態における D L S 管理テーブルの構成を示す図。

【図 1 0】

同実施形態における P L S コマンドテーブルの構成を示す図。

【図 1 1】

同実施形態における P L S コマンドに対する DMA コントローラの処理手順を示すフローチャート。

【図 1 2】

同実施形態における D L S コマンドテーブルの構成を示す図。

【図 1 3】

同実施形態における D L S コマンドに対する D M A コントローラの処理手順を示すフローチャート。

【図 1 4】

同実施形態における P E コマンドテーブルの構成を示す図。

【図 1 5】

同実施形態における P E コマンドに対するプロセッサエレメントの処理手順を示すフローチャート。

【図 1 6】

同実施形態における D M A コントローラおよびプロセッサエレメントと各テーブルとの関係を示す図。

【図 1 7】

同実施形態におけるマルチプロセッサシステム全体の動作の様子を模式的に示す図。

【符号の説明】

- 1 0 … バス
- 1 1 … マスタプロセッサ (M P)
- 1 2 - 1, 1 2 - 2 … プロセッサエレメント (P E)
- 1 3 - 1, 1 3 - 2 … プログラムローカルメモリ (P L S)
- 1 4 - 1, 1 4 - 2 … データローカルメモリ (D L S)
- 1 5 - 1, 1 5 - 2 … D M A コントローラ
- 1 6 … メモリコントローラ
- 1 7 … グローバルメモリ (G M)
- 1 1 1 … カウンタアレイ
- 1 2 1, 1 5 1 … コマンドプーリングバッファ
- 2 0 1 … プロセッサモジュール
- 2 0 2 … バスコントローラ
- 2 0 3 … カウンタアレイ
- 3 0 1, 4 0 1 … P L S 管理テーブル

3 0 2, 4 0 2 … D L S 管理テーブル

3 0 3, 4 0 3 … P L S コマンドテーブル

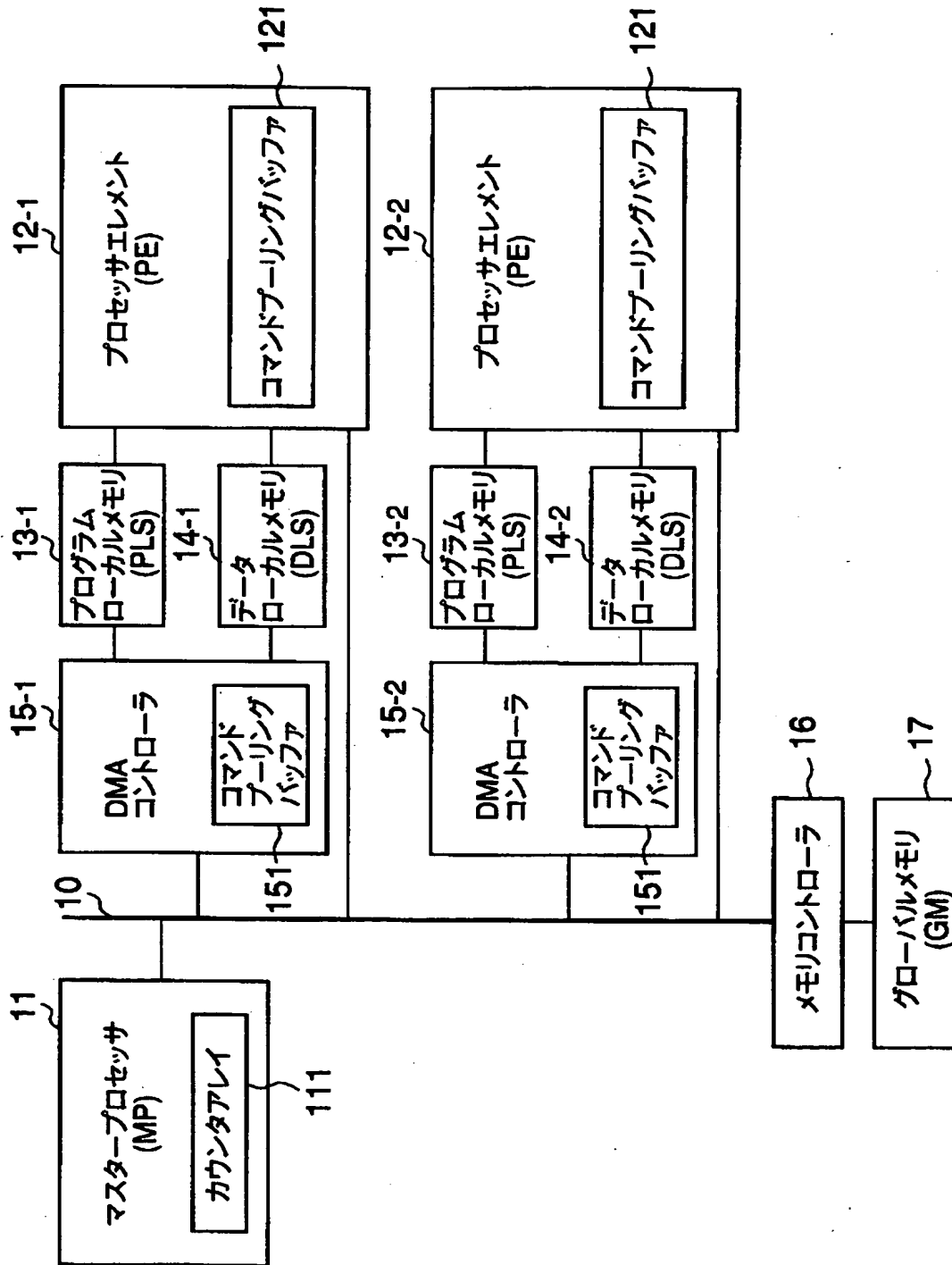
3 0 4, 4 0 4 … D L S コマンドテーブル

3 0 5, 4 0 5 … P E コマンドテーブル

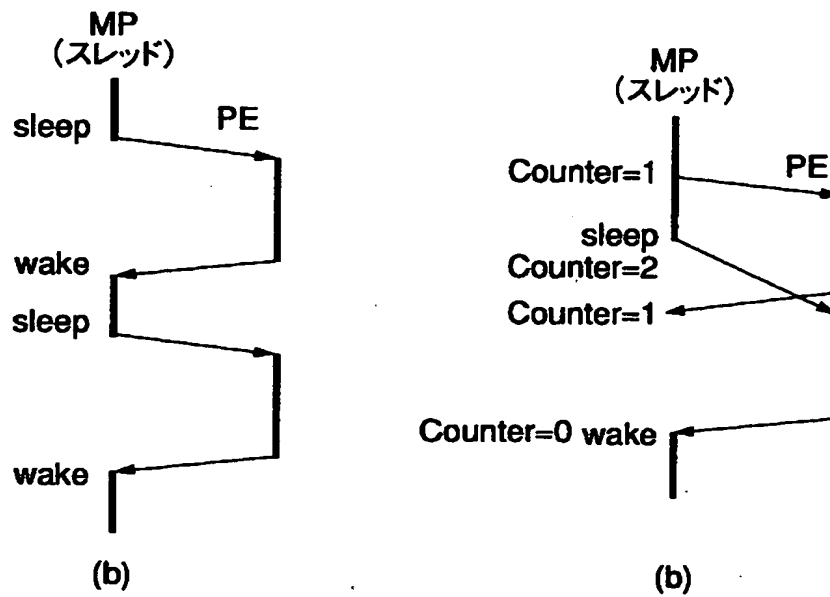
【書類名】

図面

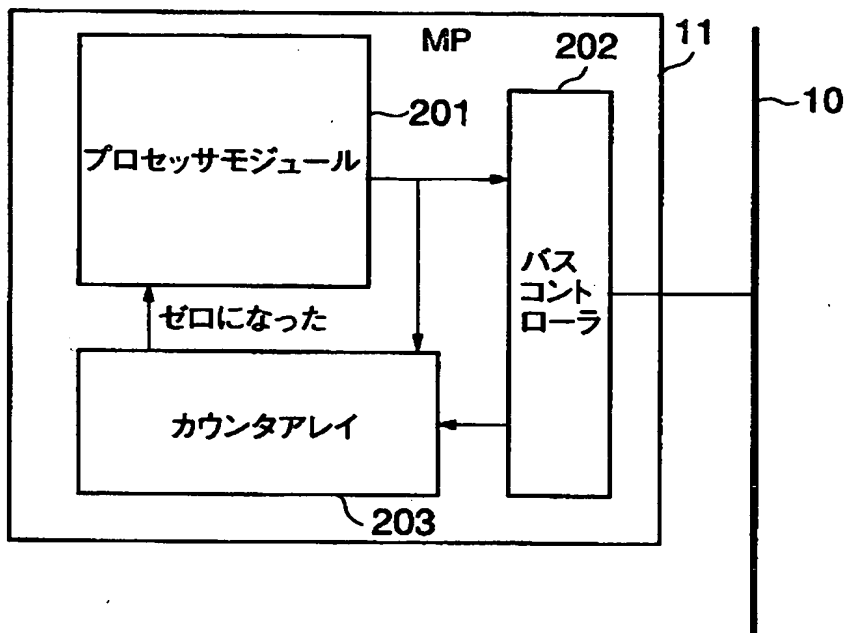
【図 1】



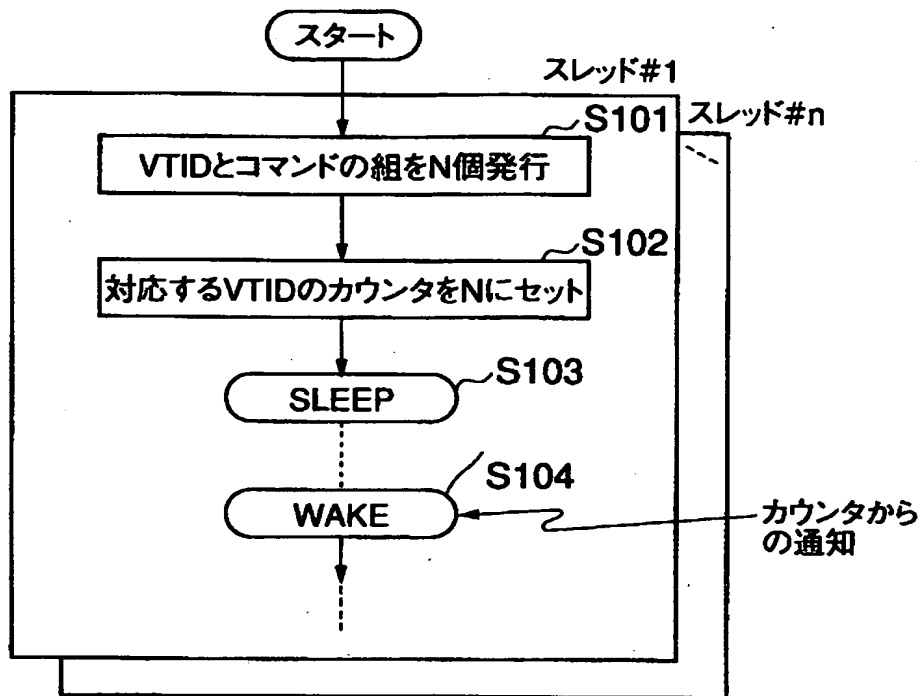
【図2】



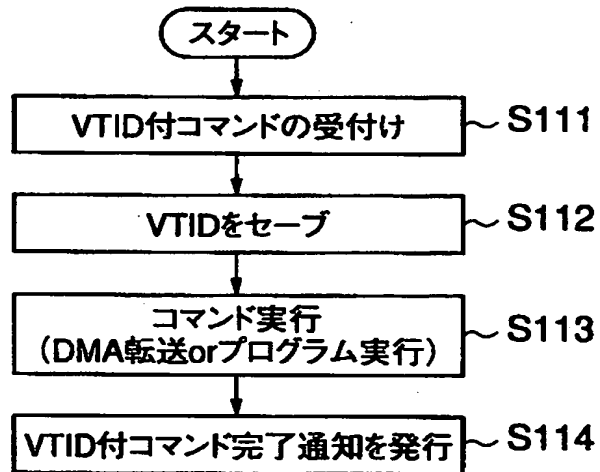
【図3】



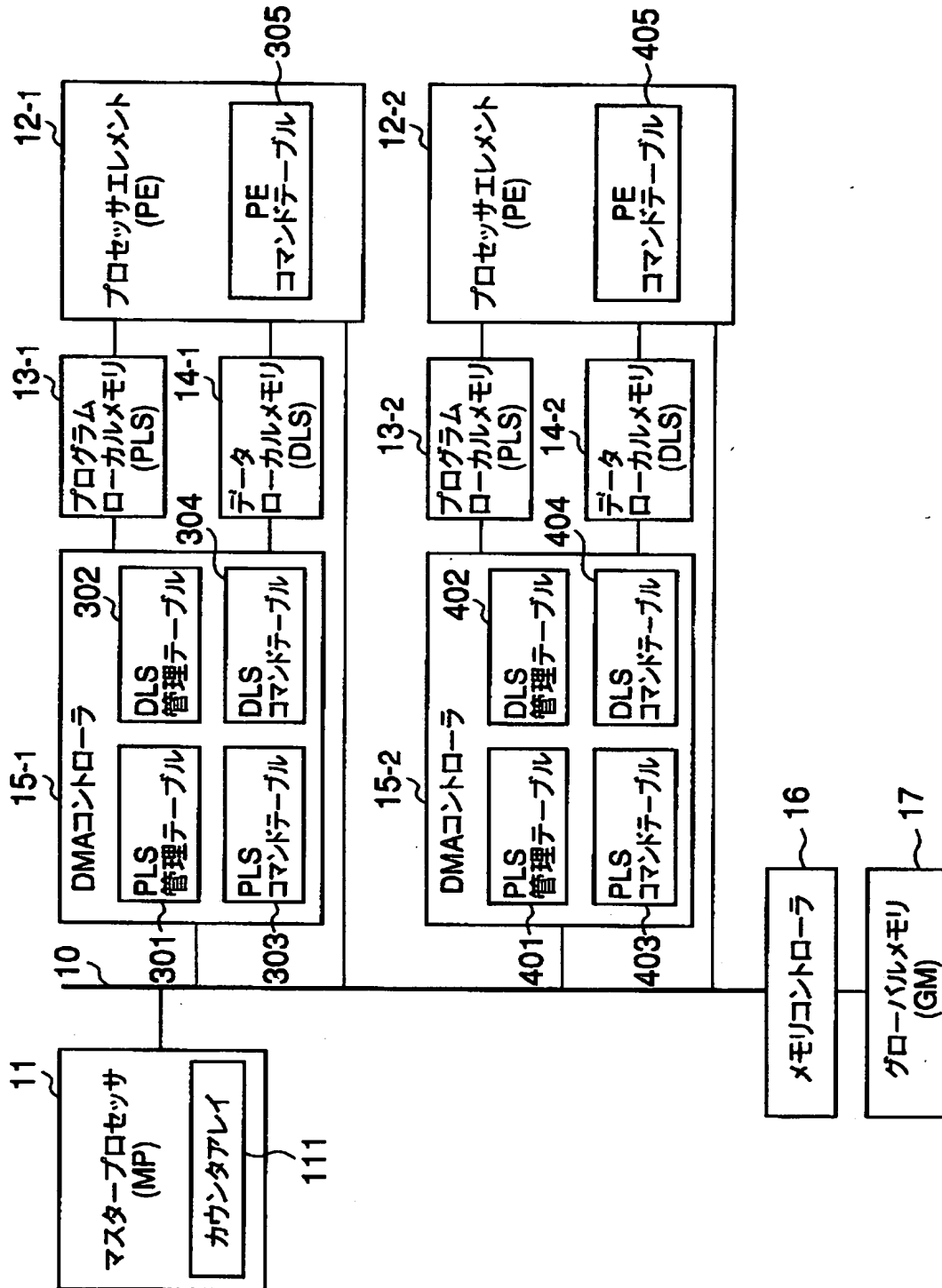
【図 4】



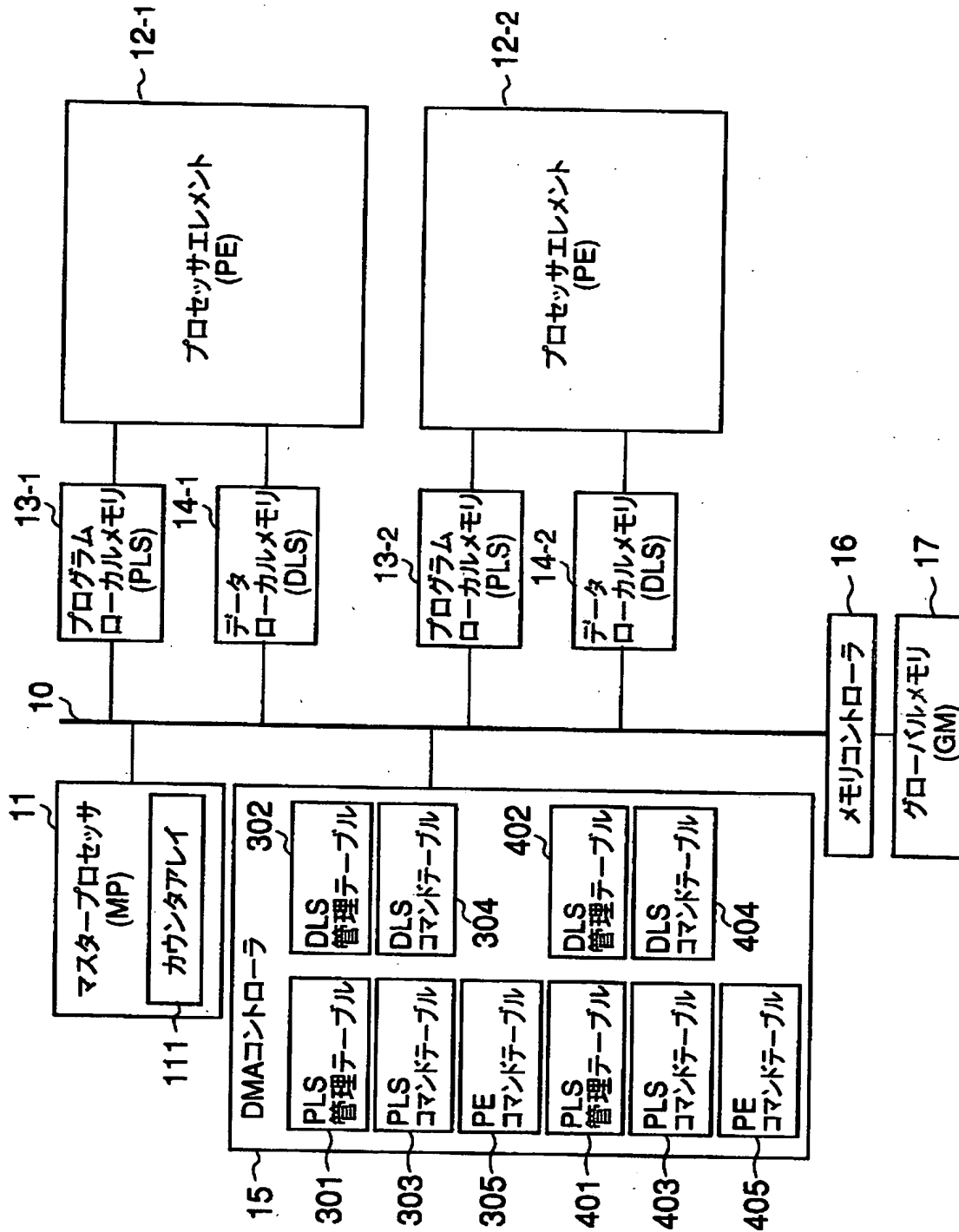
【図 5】



【図6】



【図 7】



【図 8】

PLS管理テーブル

0	44
1	44
2	44
3	—
~~~~~	
12	—
13	—
14	—
15	—

【図 9】

DLS管理テーブル

0	44	D	
1	44	D	
2	44		R
3	44		R
~~~~~			
12	—		
13	—		
14	—		
15	—		

(a)

DLS'管理テーブル

0	44	D	
1	44	D	
2	44	D	R
3	44	D	R
~~~~~			
12	—		
13	—		
14	—		
15	—		

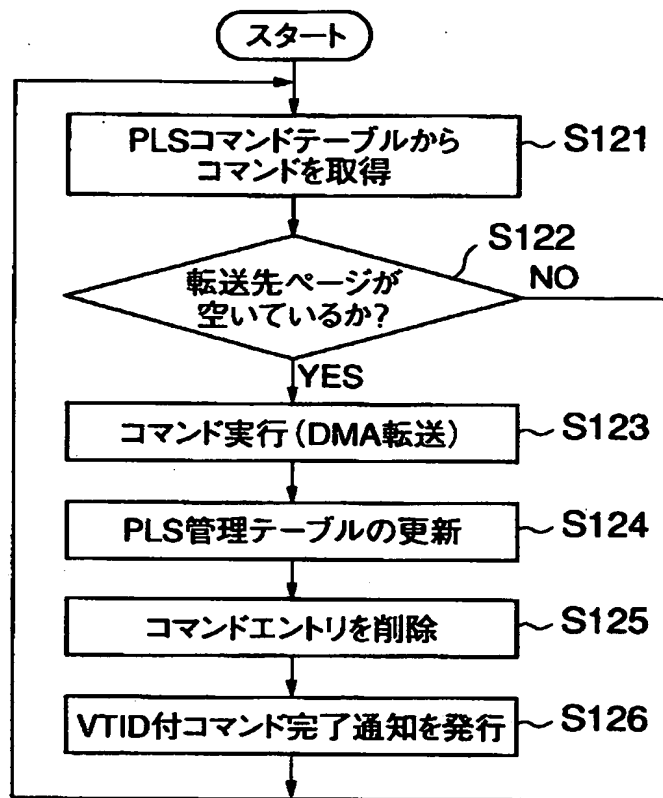
(b)

【図 1 0】

VTID	転送元GMアドレス	転送先ページ(ビットベクター)											
		0	1	2	3	.....				13	14	15	
44	0x120000	1	1	1	0	.....				0	0	0	
50	0x140000	0	0	0	0	.....				0	1	1	

PLSコマンドテーブル

【図 1 1】

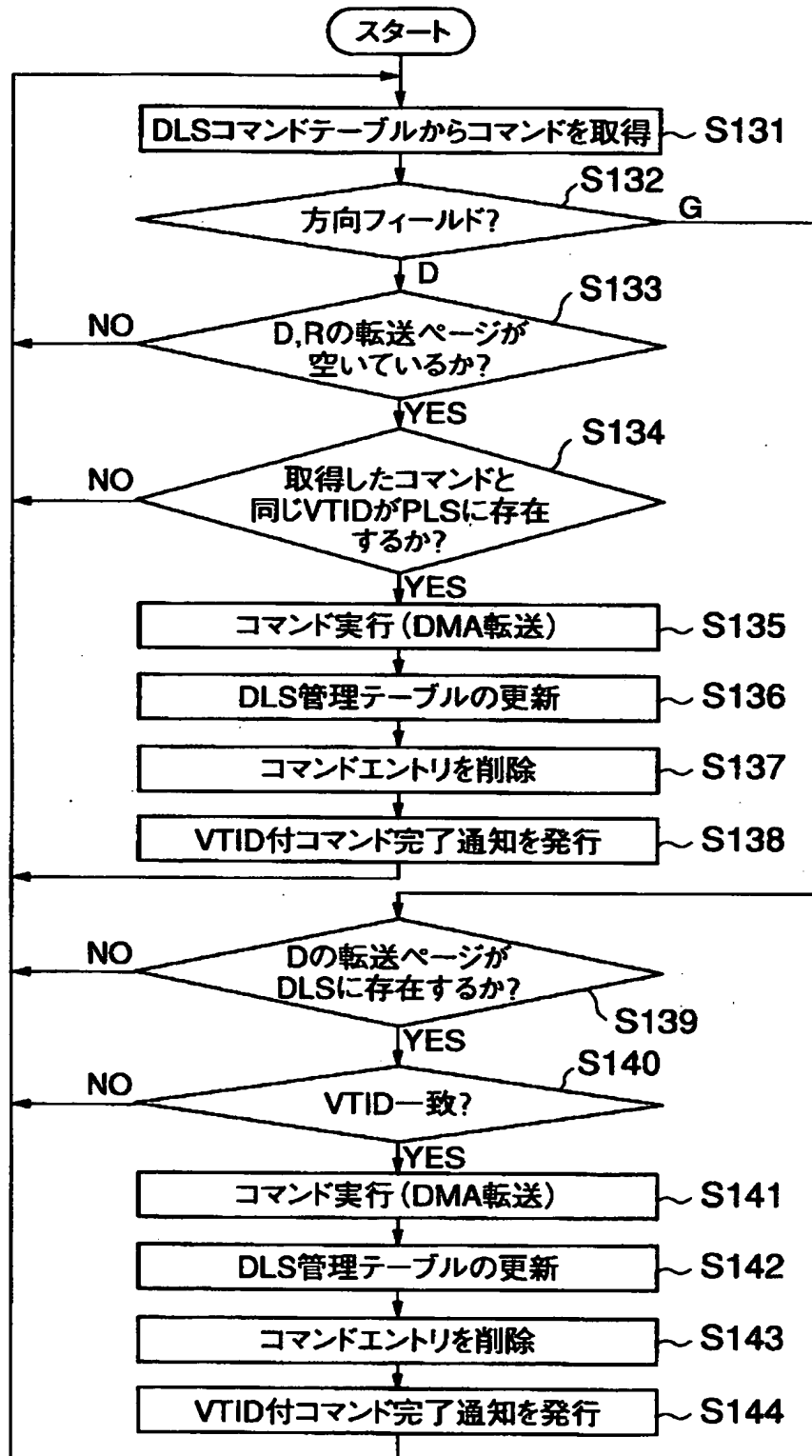


【図 1 2】

VTID	GMアドレス	方向	転送ページ(ページ毎)											
			0	1	2	3					13	14	15	
44	0x20000	D	D	D	R	R	...				0	0	0	
44	0x40000	G	0	0	D	D	...				0	0	0	

DLSコマンドテーブルの構成

【図 13】

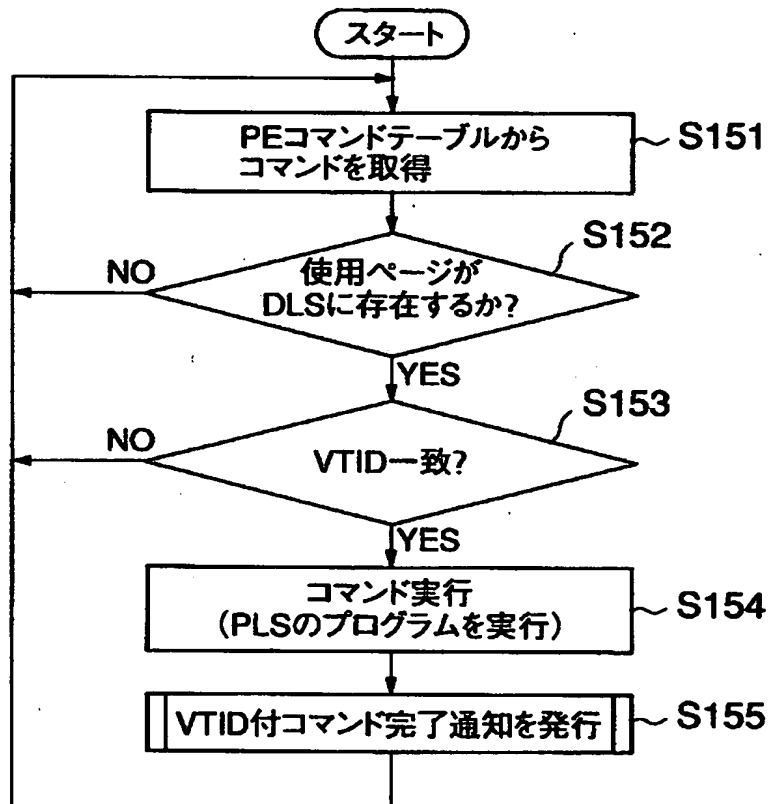


【図 1 4】

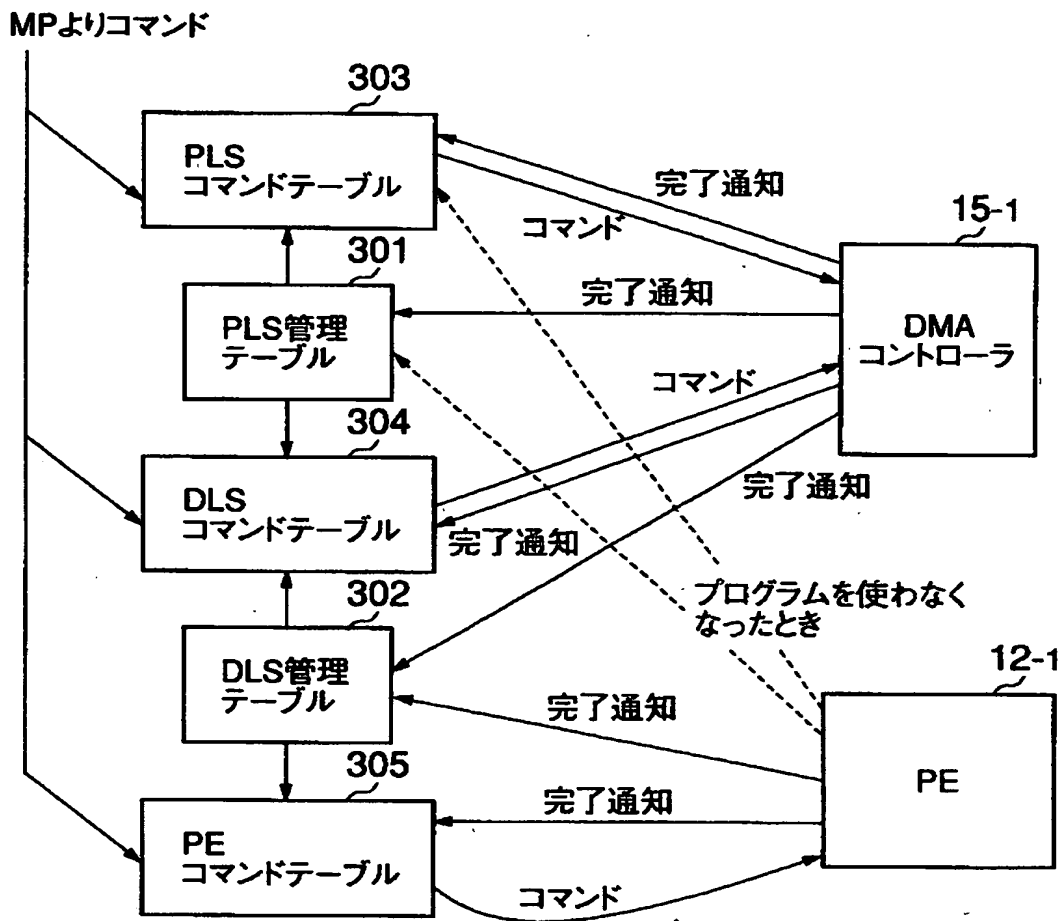
VTID	開始ページ	使用ページ(ビットベクター)											
		0	1	2	3	...			13	14	15		
44	0	1	1	0	0	...			0	0	0		
50	0	0	0	1	1	...			0	0	0		

PEコマンドテーブル

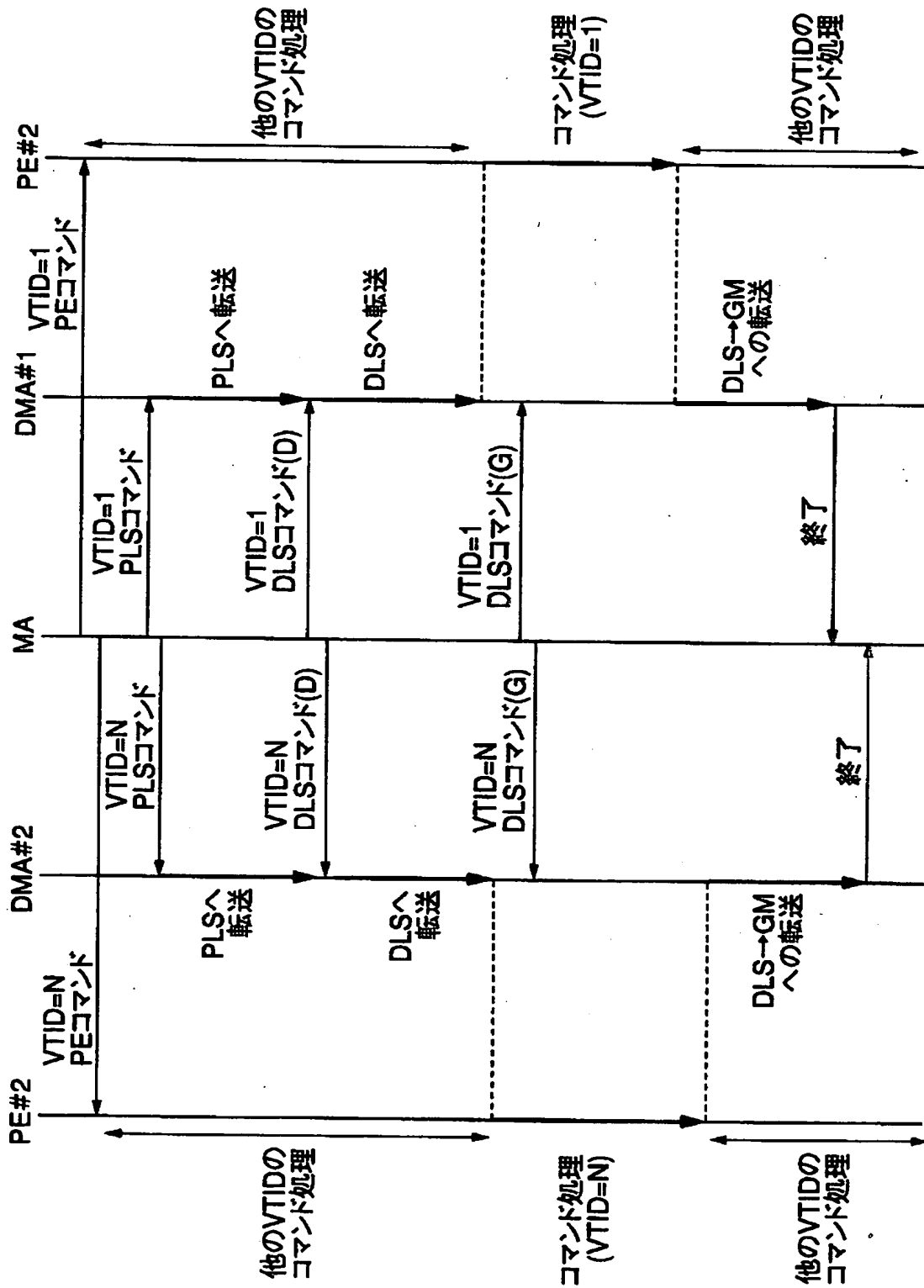
【図 1 5】



【図 16】



【図 17】



【書類名】 要約書

【要約】

【課題】 マスタプロセッサの負荷の増大を招くことなく、マルチプロセッサシステムのスループットの向上を図る。

【解決手段】 プロセッサエレメント（P E）にはそれぞれ複数のコマンドを蓄積可能なコマンドブーリングバッファ 1 2 1 が設けられ、またDMAコントローラにもそれぞれ複数のコマンドを蓄積可能なコマンドブーリングバッファ 1 3 1 が設けられている。マスタプロセッサ（M P） 1 1 からDMAコントローラや各プロセッサエレメント（P E）へのコマンドは複数まとめて発行することができ、先に送ったコマンドに対する応答を待たずに次のコマンドを発行することができる。さらに、発行済みで応答が返ってきてないコマンドの数の管理はカウンタアレイ 1 1 1 によって行われ、すべての発行済みのコマンドに対して応答が返ってきたときにそのことがマスタプロセッサ（M P） 1 1 に通知される。

【選択図】 図 1



出 願 人 履 歴 情 報

識別番号 [000003078]

1. 変更年月日 1990年 8月22日  
[変更理由] 新規登録  
住 所 神奈川県川崎市幸区堀川町72番地  
氏 名 株式会社東芝
2. 変更年月日 2001年 7月 2日  
[変更理由] 住所変更  
住 所 東京都港区芝浦一丁目1番1号  
氏 名 株式会社東芝